EXPRESS MAIL LABEL NO.: EL443489750US

Atty Dkt No.: 99 P 7817 US

APPLICATION FOR LETTERS PATENT OF THE UNITED STATES

NAME OF INVENTORS: PEIYA LIU
39 DAVISON AVENUE
EAST BRUNSWICK, NJ 08816
UNITED STATES OF AMERICA

LIANG-HUA HSU
4 ORLY COURT
ROBBINSVILLE, NJ 08691
UNITED STATES OF AMERICA

YOUNG FRANCIS DAY
35-24 QUAIL RIDGE DRIVE
PLAINSBORO, NJ 08536
UNITED STATES OF AMERICA

VIVEK AGRAWALA

TITLE OF INVENTION: **AUTOMATED GENERATION OF CARD-BASED PRESENTATION DOCUMENTS FROM MULTIMEDIA DATA**

TO WHOM IT MAY CONCERN, THE FOLLOWING IS A SPECIFICATION OF THE AFORESAID INVENTION

Automatic Generation of Card-Based Presentation Documents from Multimedia Data

Background of the Invention

Field of the Invention

The present invention relates to multimedia document presentations and more particularly to a system for automatically generating platform-independent formatting object descriptions from SGML documents and non-textual media sources which can then be used to create card-based multimedia presentation documents.

Description of the Prior Art

Traditionally, hypermedia presentations are interactively designed and manually created with hypermedia authoring tools. Various commercial hypermedia authoring tools adopt different interactive authoring paradigms but they are limited in supporting the automated generation of card-based presentation from existing multimedia document sources in large scale. The representative presentation authoring tools are summarized as follows.

PowerPoint from Microsoft is based on a structure-oriented model and supports hierarchical information content authoring in a 2D layout. Many commercial word processing tools follow this authoring model. Documents are often developed in terms of hierarchical structures such as book, pages or slides, sections, subsections, etc., and WYSWIG user interfaces are provided to support structure editing and issue formatting commands interactively.

Authorware from Macromedia and IconAuthor from AimTech are based on a flowchart model and use icons representing events such as audio or video, if-then functions, branching and hyperlinks in a linear progression like flowchart control. Content editors to assign real media files and properties to each icon could be used. This model provides high-level program control and global ordering of information presentation.

Director from Macromedia, based on a time-line model, displays media sequences as tracks and specializes in synchronizing events. It can be used to create high-level multimedia presentations.

Multimedia Toolbook from Asymetrix, based on an object-oriented model with scripting capability, provides more support of complex interaction behavior. Users can interactively enter document content (in multimedia objects) and define object properties including various formatting commands and scripts for object behavior. This authoring tool allows the user to get to a lower-level control of object and system behavior with script commands.

Summary of the Invention

The present invention provides a new approach for automatic generation of card-based document presentation from multimedia data sources. An *automatic card-based presentation generation system* programmatically creates card-based presentation documents from source documents. In the prior art, the card-based document presentation is interactively created by authoring tools. However, if documents are in large scale and information content already exists in other forms (e.g., archive formats), the interactively authoring process for manually creating presentation forms from these existing documents becomes

tedious and time-consuming. This invention includes an automated system for transforming a card layout style specification into presentation specifications which at run-time, are used to generate formatting object descriptions. The formatting object descriptions can then be used to actually create card-based presentation documents.

The present invention comprises a *presentation style transformer* and a *card-based presentation generator* for generating card-based Formatting Object Descriptions (FOD) from the Card Layout Style Specification (CLSS). The presentation style transformer takes the card layout style specification (CLSS) and the *Card Display Schema* (CDS) as input and transforms them into a *Card-based Presentation Specification* (CPS). CLSS is used to declaratively specify the layout of the card-based presentation. CDS is used to specify *meta rules* for presentation resources and for variable definitions. CPS is a procedural specification of overall card-based presentation characteristics including layout, resources and presentation procedures. The card-based presentation generator takes the card-based presentation specifications and card-based document content as input to create formatting object descriptions. The generator first uses a *Presentation Construct Mapper* for translating CPS constructs into constructs required by *Card-Based DSSSL Style Specifications* and then uses a *Card-based DSSSL Processor* to create a *Card-Based Document Flow Object Tree*. After that, the generator uses an *FOD Converter* to convert the flow object tree into formatting object descriptions.

Brief Description of the Drawings

Figure 1 illustrates a block diagram of the

automatic card-based presentation generation system of the present invention.

Figure 2 illustrates the syntax of a card layout style specification.

Figure 3 gives an example of a card layout style specification for CardTitle.

Figure 4 illustrates the syntax of a generated card-based presentation specification.

Figure 5 gives an example of a card-based presentation specification for CardTitle.

Figure 6 illustrates a block diagram of a presentation style transformer for translating card layout style specifications into card-based presentation specifications based on display schema specifications.

Figure 7. illustrates a display schema for generating a TextbyGfxTitle resource declaration.

Figure 8 illustrates a syntax of display schema for generating resource declarations or variable definitions.

Figure 9 illustrates a display schema for generating content variable definitions.

Figure 10 illustrates a display schema for generating all primitive resource declarations.

Figure 11 illustrates a card-based context tree.

Figure 12 illustrates card-based context paths.

Figure 13 illustrates a flow chart of the content mapping rule generation process.

Figure 14 illustrates examples of generated CPS for C, CT and T nodes.

Figure 15 illustrates examples of generated CPS for R and A nodes.

Figure 16 illustrates a block diagram of a card-based presentation generator for translating card-based presentation specifications into structured formatting object descriptions.

Figure 17 illustrates a generated card-based DSSSL style for CardTitle.

Figure 18 illustrates some card-based DSSSL style functions used for CardTitle.

Figure 19 illustrates a CardTitle formatting object description.

Figure 20 illustrates a general syntax of formatting object description.

Detailed Description of the Invention

This patent application is related to copending U.S. patent application entitled "A Generalized System for Automatically Hyperlinking Product Documents", Attorney Docket No. 99P7818US, filed on September 22, 1999 and assigned to the same assignee as the present invention.

The present invention is an automatic card-based presentation generation system that programmatically generates card-based formatting object descriptions from large SGML textual documents and non-textual media sources. The formatting object descriptions can be used to automatically create card-based presentation document formats. The prior art of the presentation approach is based on interactive authoring tools for manually creating multimedia presentation. The interactive approach does not support creating large-scale presentation documents for industrial applications due to required human involvement in the authoring process.

The overall card-based presentation generation system is described in Figure 1. The generation system comprises presentation style transformer 12 and card-based presentation generator 14. Presentation style transformer 12 receives a card layout style specification (CLSS), examples of which are shown in Figures 2 and 3. The CLSS language is described in US Patent Application

serial number 08/984,734 filed on December 4, 1997, entitled "Style Specifications For Systematically Creating Card-Based Hypermedia Manuals" and assigned to the same assignee as the present invention. Presentation style transformer 12 also receives a card display schema, an example of which is shown in Figure 8. Presentation style transformer 12 generates a card-based presentation specification, shown in Figures 4 and 5, as output. The generated card-based presentation specification comprises two parts: MACRO resource declarations and procedural element mapping rules.

Card-based presentation generator 14 receives both, the card-based presentation specification and the SGML document content as inputs and generates platform-independent formatting object descriptions, FODs, as output. Formatting object descriptions are abstract descriptions of formatting directives of presentation documents. Once FODs are created, an automatic scripting process can be used to generate a card-based document presentation. The automatic scripting process is described in US Patent Application serial number 08/986,270 filed on December 5, 1997, entitled "Formatting Card-Based Hypermedia Documents By Automatic Scripting" and assigned to the same assignee as the present invention.

Presentation style transformer, 12 of Figure 1, is shown in greater detail in Figure 6. The presentation style transformer comprises two major components: resource generator 22 and style proceduralizer 24. Resource generator 22 is based on the card display schema and creates presentation resource declarations from the card-based layout style specification. Card presentation resource declarations include two types of resources: MACRO_*Resource*(*stylename, parameters*) for primitives and MACRO_*name*(*stylename, parameters*) for composites. The

first argument of a MACRO resource declaration must be a stylename. Primitive resources are used for basic presentation object resource requirements of the target presentation platform. Composite resources can be defined in terms of primitive ones or other composite resources.

Resource generator 22 creates MACRO resource declarations by examining the objects inside background list and background list of card-based style specifications and the corresponding card display schema for the objects. As an example, the generated CardTitle MACRO resource declaration is shown in Figure 5. This generation procedure is triggered by the TextByGfxTitle style layout specification shown in Figure 3. The corresponding display schema, shown in Figure 7, is used to guide the generator to create the TextByGfxTitle resource declaration with default values of their parameters. The MACRO_TitleField specifies the needed presentation objects and the variable $CardTitle_RTF$ whose value would be calculated by a predefined DSSSL function RTF-TitleGen from source document content at run-time.

The display schema syntax is shown in Figure 8. It can be used to guide the generator for both creating resource declarations in the <MACRO_Declaration> part and variable definitions in the <ContentMapping> part of the card-based presentation specification. Display schema for generating a variable definition of $CardTitle_RTF$ is done by style proceduralizer 24, shown in Figure 9. Display schema for generating all primitive resource declarations is shown in Figure 10.

Style proceduralizer 24, in Figure 6, comprises three components: context tree builder 26, content node path walker 28 and content mapping rule generator 30. Context tree builder 26 creates a context tree for rule

generator 30 to create content mapping rules along content node paths in the tree. Style proceduralizer 24 translates a declarative card layout style specification into procedural card content mapping rules. For example, CardTitle element content mapping is shown in Figure 5. The first element mapping rule procedurally specifies the following things: Any document content with tag CardTitle or ANYDOCX under document tag CARDX which is under tag CARD will use an element rule with matched context tags for creating formatting object descriptions. Since the second element rule matches the context tags (Card, CardX, CardTitle), it will be invoked for further specifying how to create formatting object descriptions for CardTitle. The second rule basically specifies which types of formatting objects are created by using a stylename attribute value under a different cardtype.

Context tree builder 26 takes all context information in the card layout style specifications as input and constructs a context tree as output by examining and ordering all context tags appearing in the context and AIU attribute values in the layout specifications, by grouping the same context specification into one with type differences, and by marking context tags into the following categories: R (root node), C (content node) for both being a leaf node of context tree and the last element of some context attribute value, T (transition node) for both being a non-leaf node and non-content node, CT (content/transition node) for both being a non-leaf node of context tree and the last element of some context attribute value, and A (associated node) from attribute AIU values for the context of associated AIU in graphics. The context tree is designed to capture content mapping rule context for making an efficient generation process

of procedural rule mappings in CPS. As an example, a context tree is shown in Figure 11.

Content node path walker 28 takes the context tree as an input and finds all context paths as the output. The context paths are defined to be the shortest paths from the root node to each C, CT and A nodes as shown in Figure 12.

Content mapping rule generator 30 uses categories and types of nodes to generate constructs of content mapping rules. The detailed process in shown in Figure 13. Examples are shown in Figures 14 and 15.

For transition nodes, mapping rule generator 30 adds element rule constructs which simply specify how to pass rule processing to their children in the context tree. For content nodes, the mapping rule generator adds specifications of the formatting object descriptions. For content transition nodes, it adds rule specifications for both content and transition functionality. For root and associated nodes, the mapping rule generator adds predefined and domain-specific presentation specifications. Particularly for the root node, the rule generator adds the specifications of defining an overall card-based presentation structure. For associated nodes, the mapping rule generator adds structure-specific association specifications. For instance, the structure-specific specifications can be related to hotspots, annotations, or animation objects associated with graphic objects.

Card-based document presentation generator (14 of Figure 1) shown in Figure 16 comprises the following components: presentation construct mapper 42, card-based DSSSL processor 44 and FOD convertor 46.

Presentation construct mapper 42 is used to map constructs in CPS into Card-based DSSSL constructs shown in Figures 17 and 18. The mapping procedure is a one-

pass translation from CPS to a card-based DSSSL style specification as follows. Card-based DSSSL style specification uses the ISO standard document style language, called DSSSL (Document Style Semantics and Specification Language).

(1) Map CPS ElementRule context into card-based DSSSL element context.

(2) Map CPS CaseExpr and IfExpr into card-based DSSSL *case* and *if* functions.

(3) Map CPS MACRO_Declaration into card-based predefined DSSSL functions.

(4) Map CPS ProcessGrove into card-based DSSSL *Process* function.

(5) Map CPS DefineVar, AddObject and AddGroup into card-based DSSSL FODfo flow object instance.

This mapping process is very efficient, since CPS is designed to be as close to card-based DSSSL style specification as possible. In particular, the card-based DSSSL style specifications are based on one universal card-based presentation flow object, called FODfo, for creating the card-based document flow object tree as shown in Figure 16. A card-based document flow object tree consists of a sequence of FODfo object instances. It is an abstract representation of card-based document formatting object descriptions. Using this single universal flow object can greatly simplify mapping of constructs from CPS to card-based DSSSL style specifications.

To create an instance of FODfo flow object, the mapper only needs to make a DSSSL function *makeFODfo* call, which takes three keyed arguments: FOtype, FOname and FOinstruction. FOinstruction is a data string that represents formatting information such as the presentation object attribute values. The key concept here is that FODfo flow object is designed in such a way

that a complete card-based formatting object description can be composed by appending a sequence of FODfo object instances. Thus, a FOD can be created by a DSSSL predefined function *sosofo-append* (specification of sequence of flow object-append) to append all FODfo instances.

Card-based DSSSL processor 44 is used to create a card-based document flow object tree. This is an abstract representation of card-based document formatting descriptions and it comprises a sequence of FODfo flow objects. Each FODfo flow object contains an SGML data string of FOD and each FODfo flow object is designed for uniformly representing card-based formatting object information.

Finally, FOD converter 46 is used to convert this abstract representation of sequences of FODfo objects into FODs, i.e., formatting object descriptions, shown in Figures 19 and 20.

The card-based presentation specification is designed for bridging the gap between the declarative card layout style specification and the procedural card-based DSSSL style specification. These specifications can be automatically generated from card layout style specifications and can be used for automatically generating card-based DSSSL style specifications.

It is not intended that this invention be limited to the hardware or software arrangement or operational procedures shown disclosed. This invention includes all of the alterations and variations thereto as encompassed within the scope of the claims as follows.